Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)
факультет інформаційних технологій
(факультет)
Кафедра інформаційних систем та технологій та комп'ютерної інженерії
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА

кваліфікаційної роботи ступеня магістра

(бакалавра, спеціаліста, магістра)

Студента Бородіна Ігора Анатолійовича
(ПІБ)
Академічної групи 126м-20-1
(шифр)
спеціальності 126 «Інформаційні системи та технології»
(код і назва спеціальності)
за освітньо-професійною програмою

«Інформаційні системи та технології»
(офіційна назва)

на тему Розробка інформаційної технології оцінки точності Web-сервісів прогнозу опадів
(назва за наказом ректора)

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи	доц. Гаркуша І.М.			
розділів:				

Рецензент	Довбніч М.М.			

Нормоконтролер	проф. Коротенко Г.М.			

Дніпро

2022

ЗАТВЕРДЖЕНО:
завідувач кафедри
інформаційних технологій
та комп'ютерної інженерії
(повна назва)
_____ Гнатушенко В.В.
(підпис) (прізвище, ініціали)
«_____»_____2021_року

ЗАВДАННЯ

на кваліфікаційну роботу
ступеня магістр

(бакалавра, спеціаліста, магістра)

студенту Бородіну І. А. **академічної групи** 126м-20-1
(прізвище та ініціали) (шифр)

спеціальності 126 « Інформаційні системи та технології »

за освітньою-професійною програмою «Інформаційні системи та технології»

на тему Розробка інформаційної технології оцінки точності Web-сервісів прогнозу опадів

затверджену наказом ректора НТУ «Дніпровська політехніка» від _____№ ____

Розділ	Зміст	Термін виконання
Розділ 1	Аналіз стану області рішення задачі	1.10.2021 – 31.10.2021
Розділ 2	Моделі та методи розв'язання задачі	1.11.2021 – 30.11.2021
Розділ 3	Програмна реалізація розробленої інформаційної системи	1.12.2021 – 21.12.2021

Завдання видано _____ Гаркуша І.М.

Дата видачі 1.10.2021 р.

Дата подання до екзаменаційної комісії _____

Прийнято до виконання _____ Бородін І.А.

РЕФЕРАТ

Пояснювальна записка: 45 стор., 8 рис., 3 табл., 4 додатки, 8 джерел.

Об’єкт дослідження: точність надання інформації щодо передбачення опадів.

Предмет дослідження: інформаційні технології порівняння передбачених опадів з фактичними.

Мета магістерської роботи: створення застосунку для опрацювання даних сервісів передбачення опадів на базі фреймворка Spring, застосунку для опрацювання GPM даних у HDF5 представленні на мові програмування Java .

У вступі розглядається аналіз та сучасний стан проблеми, конкретизується мета даного дослідження.

У першому розділі проаналізовано предметну галузь, визначено актуальність завдання та призначення розробки, наведено огляд існуючих рішень.

У другому розділі наводяться отримані метричні дані щодо опадів та проводиться їх аналіз.

Третій розділ присвячено проектуванню, розробці та реалізації інформаційних систем опрацювання даних опадів.

Практична значимість полягає у створенні додатка, який дозволить проаналізувати GPM дані, а також додатка що надаю змогу групувати дані надані іншими Web-сервісами.

Актуальність інформаційної системи визначається великою кількістю сервісів прогнозування опадів, однак у той самий час відсутня інформація наскільки точні метрики вони надають.

Список ключових слів: СЕРВІС, API, ОПАДИ, HDF5, JAVA, GPM.

ABSTRACT

Explanatory note: 45 pages, 8 figures, 3 tables, 4 app., 8 sources.

Object of research: accuracy of providing information regarding rainfall forecasting.

Subject of research: information technologies for comparing predicted precipitation with actual.

The goal of master's work: creation of an application for data processing of precipitation forecasting services based on the Spring framework, an application for processing GPM data in HDF5 presentation in the Java programming language.

The introduction discusses the analysis and current state of the problem, specifies the purpose of this study.

The first section analyzes the subject area, determines the relevance of the task and purpose of development, provides an overview of existing solutions.

The second section presents the obtained metric data on precipitation and analyzes them.

The third section is devoted to the design, development and implementation of information systems for data processing of precipitation.

The practical significance lies in the creation of an application that will analyze GPM data, as well as an application that allows you to group data provided by other Web-services.

The relevance of the information system is determined by the large number of precipitation forecasting services, but at the same time there is no information on how accurate metrics they provide.

Keywords: SERVICE, API, PRECIPITATION, HDF5, JAVA, GPM.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ІС – інформаційна система

ПЗ – програмне забезпечення

HTTP – Hyper Text Transfer Protoco

REST – Representational State Transfer — «передача стану уявлення» — архітектурний стиль взаємодії компонентів

EOSDIS –The Earth Observing System Data and Information System (EOSDIS)

GPM – Global Precipitation Measurement

HDF5 – Hierarchical Data Format 5

URL – Uniform Resource Locator

WWO – WorldWeatherOnline

ВСТУП

Атмосферні опади є однією з характеристик клімату, інтерес до якої ніколи не слабшає. Ставлячись до явищ, вплив яких на діяльність людини і навколишнє його середовище найбільш велике і може проявлятися як в позитивному, так і в негативному сенсі, вони постійно привертають увагу екологів, гідрологів і метеорологів. Рясні опади часто ускладнюють роботу транспорту, завдають шкоди сільському господарству і іншим сферам економіки.

Клімат змінюється, і це відбувається прямо зараз. Мова йде не про віддалене явище, яке матиме місце коли-небудь в майбутньому, і не тільки про підвищення температури. Очікується, що в деяких частинах світу річний рівень опадів в довгостроковій перспективі знизиться, в той час як в інших регіонах коливання рівня опадів і температури помітно вплинуть на вегетаційний період деяких рослин. В інших місцях річна кількість опадів може залишитися незмінним, але випадати вони можуть з великими інтервалами, у вигляді набагато більш сильних і короткочасних злив, що викликають посилення посух і повеней. Може зрости інтенсивність сильних штормів і їх різновиди - ураганів.

Завчасне та коректне прогнозування кількості опадів є запорука успіху для запобігання негативного впливу зміни клімату. Враховуючи вищеперераховане, можна стверджувати що на сьогодні оцінка точності прогнозу опадів є актуальною темою, оскільки існує велика кількість сервісів що надають прогнози, які можуть відрізнятися.

Мета даного дослідження є порівняння спрогнозованої web-сервісами кількості опадів з фактичною кількістю за допомогою реалізованого програмного додатку, який дозволяє порівнювати дані завантажені з Earthdata порталу, та даними прогнозів що надають побічні сервіси, зокрема WorldWeatherOnline.

РОЗДІЛ 1

АНАЛІЗ СТАНУ ОБЛАСТІ РІШЕННЯ ЗАДАЧІ

1.1. Загальні відомості з предметної галузі

1.1.1 Зміна клімату Землі

Хоча клімат Землі коливався і раніше, в останні 100 років це відбувається незрівнянно частіше. При цьому середня приземному температура зросла приблизно на 0,6-0,7oC. Може здатися, що це не так багато, але з тих пір як клімат став «нелінійної» динамічною системою, навіть незначні зміни температури можуть стати причиною цілого ряду каскадних наслідків. Згідно з даними інструментальних спостережень (ведуться з 1850 року), вісім найбільш теплих років були зафіксовані в період з 1998 року, а найтеплішим був 2005 рік.

Це відбувається в результаті діяльності людини. Використання нами таких видів палива, як нафта, вугілля і газ, а також вирубка лісів призвело до значного збільшення вмісту в земній атмосфері вуглекислого газу (CO2), а також інших парникових газів. Ці парникові гази створюють ефект утримання тепла (звідси назва), не дозволяючи йому йти в атмосферу. У зв'язку з тим, що парниковий ефект - це нормальне природне явище, ми назвали його «неконтрольованим парниковим ефектом», що є однією з причин глобального потепління. З часу Промислової революції, що відбулася в кінці XVIII століття, вміст CO2 в атмосфері в результаті діяльності людини значно зросла, і на сьогоднішній воно на такому рівні, що не відзначався принаймні не протязі 800 000 років.

Діяльність людини призводить до викидів чотирьох основних парникових газів: вуглекислого газу (CO2), метану (CH4), закису азоту (N2O). Ці гази накопичуються в атмосфері, викликаючи з часом підвищення концентрації. В індустріальну епоху мали місце значні збільшення концентрації всіх цих газів (див. Рис. 1.1). Всі ці збільшення можна віднести на рахунок діяльності людини.

Зростання викидів вуглекислого газу став результатом розширення використання викопних видів палива у транспортній галузі, в опаленні і охолодженні будівель, у виробництві цементу та іншої продукції. Збезлісення призводить до вивільнення CO2 і скорочує його поглинання рослинами. Вуглекислий газ також вивільняється в природних процесах, таких як гниття рослинної маси.

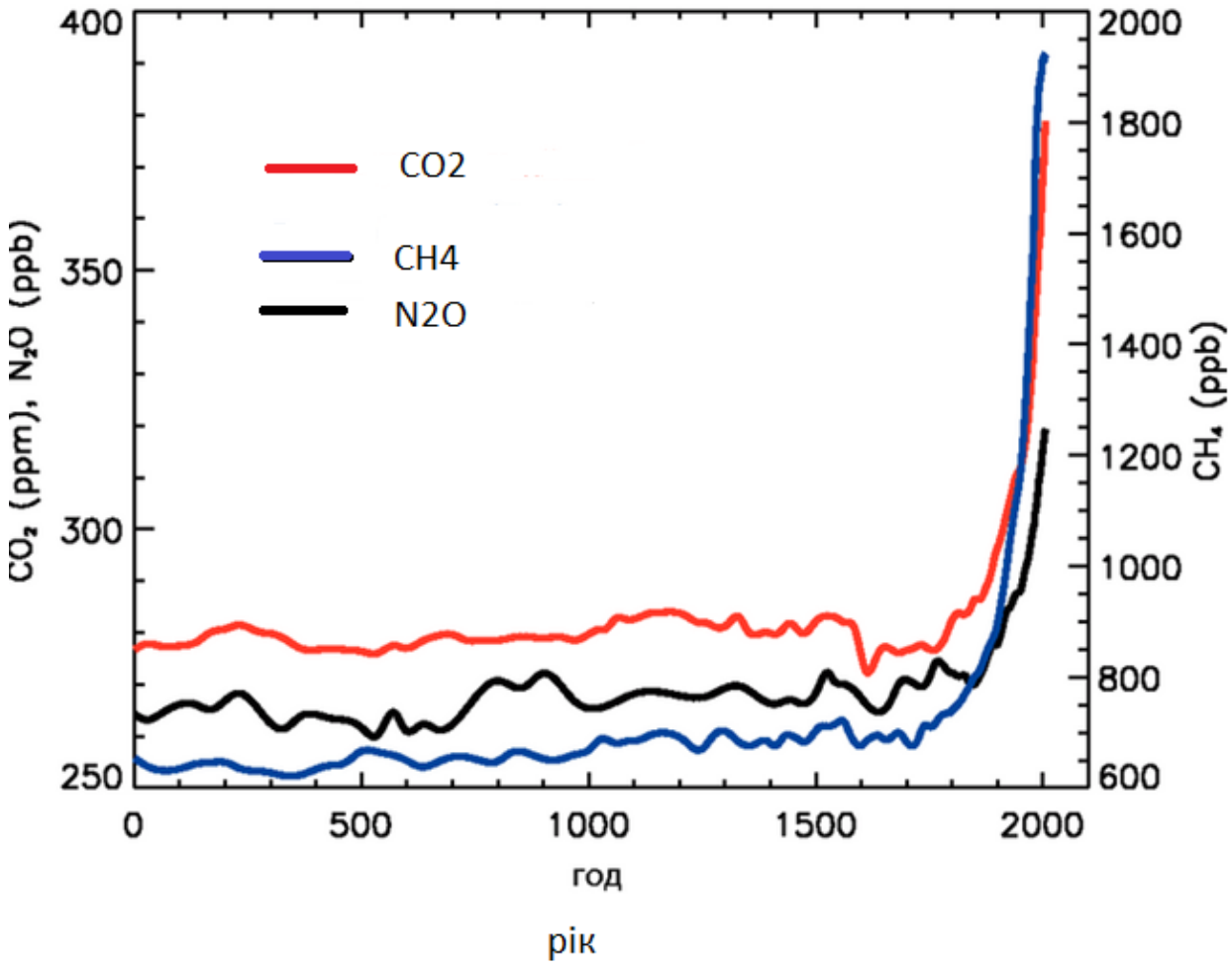


Рис. 1.1 Концентрація важких довготривалих парникових газів за останні 2000 років.

Викиди метану збільшилися в результаті діяльності людини, пов'язаної з сільським господарством, розподілом природного газу та діяльністю звалищ. Метан також вивільняється при природних процесах, які відбуваються, наприклад, на водно-болотних угіддях. В атмосфері концентрація метану зараз не збільшується, так як за останні два десятиліття темпи зростання знизилися.

Викиди закису азоту також є наслідком діяльності людини, зокрема, застосування добрив і спалювання викопних видів палива. N2O також вивільняється в ході природних процесів в ґрунтах і океанах.

Озон - це парниковий газ, який безперервно виробляється і знищується в атмосфері в ході хімічних реакцій. У тропосфері кількість озону збільшилася внаслідок діяльності людини, завдяки викидам таких газів, як окис вуглецю, вуглеводні і окис азоту, в ході хімічної реакції яких утворюється озон. Як згадувалося вище, Галоідовуглеводні, що утворюються в процесі діяльності людини, руйнують озон в стратосфері і вже привели до утворення озонової діри над Антарктидою.

Найпоширеніший і важливий парниковий газ в атмосфері - водяний пар. Проте, діяльність людини надає лише незначне прямий вплив на зміст водяної пари в атмосфері. Якщо говорити про непрямий вплив, то люди можуть істотно впливати на кількість водяної пари шляхом зміни клімату. Наприклад, більш тепла атмосфера містить більше водяної пари. Діяльність людини також впливає на водяну пару за допомогою викидів CH4, так як CH4 піддається в стратосфері хімічного руйнування, виділяючи незначна кількість водяної пари.

Зміна клімату - одне з питань розвитку. З огляду на можливого впливу цього явища на багато аспектів життя людини, сьогодні, ймовірно, воно являє собою один з найважливіших питань розвитку. Багаті країни, які давно входять до числа промислово розвинених, несуть основну відповідальність за виникнення проблеми зміни клімату, в той час як найбідніші громади і країни найбільше страждають від

наслідків, оскільки, як правило, саме вони приймають на себе головний удар сильних повеней, засух та інших передбачуваних явищ, засобів на ефективну боротьбу з якими у них не вистачає. По суті, через зміни клімату, що залишає людей в злиднях, можна втратити те, чого вдалося домогтися в сфері світового розвитку. Наприклад, зміна клімату, викликане діяльністю людини, може призвести до таких наслідків:

- негативний вплив на сільське господарство в тропіках і субтропіках (загроза продовольчій безпеці);

- подальше зменшення кількості води і погіршення її якості в регіонах, де бідні громади залежать від дощової води, використовуваної для поливу зернових і для пиття;

- поширення малярії, лихоманки денге та інших хвороб в тропічних і субтропічних регіонах (там, де охорона здоров'я і без того погано розвинене, відбудеться підвищення рівня смертності);

- буде завдано збитків екологічним системам і біологічного розмаїття в них (що спричинить за собою скорочення можливостей щодо обслуговування, забезпечення засобів до існування і доходів).

1.1.2 Прогнозування опадів

Кількісний прогноз опадів (QPF) - очікувана кількість розплавлених опадів, накопичених за вказаний період часу у зазначеній області. QPF буде створений, коли очікується, що кількість опадів досягне мінімального порогового значення протягом періоду дії прогнозу. Дійсними періодами прогнозів опадів є зазвичай синоптичні годинник, такі як 0000 0600, 1200 і 18:00 GMT. Рельєф враховується в QPF з використанням топографії або на основі кліматологічних режимів випадання опадів за результатами спостережень з високою деталізацією (Рис. 1.2).

Опади - це загальний термін для дощу, снігу та інших форм замерзлої або рідкої води, яка випадає з хмар. Кожен з цих видів опадів приносить якусь кількість вологи. Для того щоб оцінити, скільки випаде, або випало вологи, застосовується поняття «кількість опадів». Опади - переривчасте явище, і характер опадів, коли вони випадають, сильно залежить від температури і синоптичної обстановки. Остання визначає надходження вологи з вітром і через поверхневе випаровування, а також те, як вона збирається в бурю у вигляді хмар. Опади утворюються в міру конденсації водяної пари, зазвичай в висхідному повітрі, який розширюється і тому остигає. Висхідний рух обумовлено підйомом повітря через гори, рухом теплого повітря над виглядав холоднішим (теплим фронтом), вштовхування більш холодного повітря під тепліший (холодним фронтом), конвекцією від місцевого нагрівання поверхні, іншими погодними і хмарними системами.

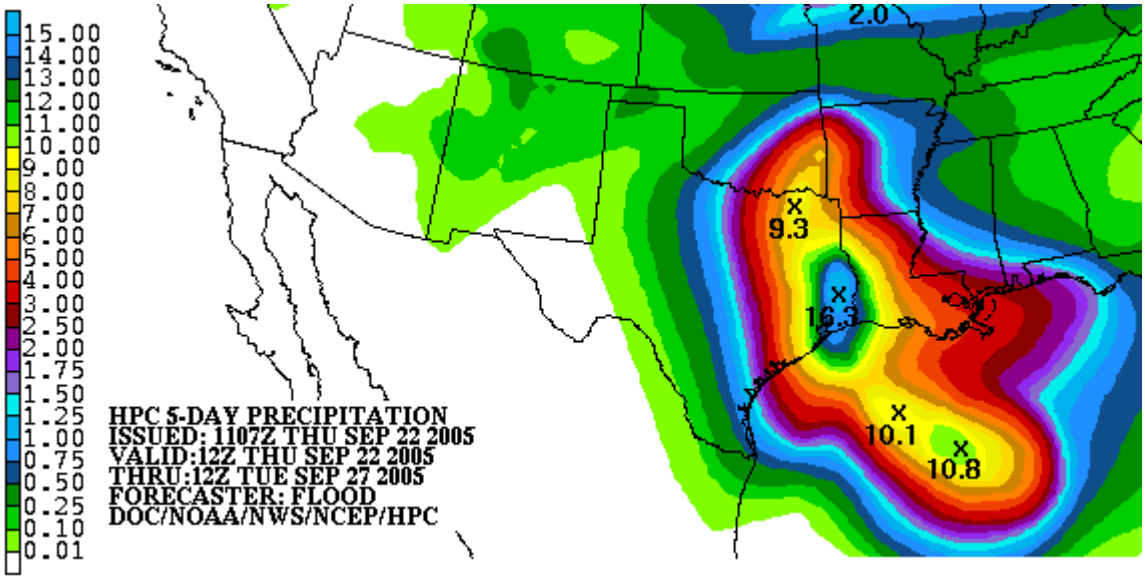


Рис. 1.2. Приклад п'ятиденного прогнозу опадів

У минулому синоптик відповідав за створення весь прогноз погоди на основі наявних спостережень. Сьогодні внесок метеорологів, як правило, обмежується вибором моделі на основі різних параметрів, таких як зміщення моделі і

продуктивність. Використання консенсусу моделей прогнозу, а також членів ансамблю різних моделей може допомогти зменшити помилку прогнозу. Однак, незалежно від того, наскільки малої стає середня помилка для будь-якої окремої системи, великі помилки в рамках будь-якого конкретного керівництва все ще можливі при будь-якому прогоні моделі. Професіонали повинні інтерпретувати дані моделі в прогнози погоди, зрозумілі для непрофесіонала. Фахівці можуть використовувати знання про місцеві ефекти, які можуть бути занадто малі за розміром, щоб їх дозволила модель, щоб додати змісти до прогноз. Наприклад, в процесі QPF враховується рельєф місцевості з використанням топографії або кліматологічних моделей опадів за результатами спостережень з високою деталізацією. Використання моделі і порівняння різних полів прогнозу з кліматології, екстремальні явища, такі як надмірні опади, пов'язані з більш пізніми повенями, дозволяють робити більш точні прогнози. Хоча підвищення точності моделей прогнозу означає, що люди можуть більше не знадобитися в процесі прогнозування в якийсь момент в майбутньому, в даний час все ще існує потреба у втручанні людини.

Прогнозування опадів протягом наступних шести годин часто називають прогнозом поточної погоди. В цьому часовому діапазоні можна прогнозувати більш дрібні особливості, такі як окремі зливи і грози, з розумною точністю, а також інші особливості, які занадто малі, щоб їх можна було вирішити за допомогою комп'ютерної моделі. Людина, що отримав останні дані радара, супутників і спостережень, зможе краще проаналізувати дрібномасштабні присутні особливості і, таким чином, зможе зробити більш точний прогноз на наступні кілька годин.

Деталізація, яка може бути включена в прогноз, зменшується з часом у міру збільшення цих помилок. Настає момент, коли помилки настільки великі, що прогноз не має кореляції з фактичним станом атмосфери. Розгляд однієї моделі прогнозу не показує, наскільки ймовірно, що цей прогноз буде правильним.

Ансамблевий прогнозування тягне за собою складання безлічі прогнозів, щоб відобразити невизначеність в початковому стані атмосфери (через помилки в спостереженнях і недостатньою вибірки). Потім невизначеність прогнозу можна оцінити за діапазоном різних прогнозів. Ансамблеві прогнози все частіше використовуються для оперативного прогнозування погоди (наприклад, в Європейському центрі середньострокових прогнозів погоди (ECMWF), Національних центрах екологічного прогнозування (NCEP) і Канадському центр прогнозування). Ансамблеві середні прогнози опадів мають ті ж проблеми, що і їх використання в інших областях, оскільки вони усереднюють більш екстремальні значення і тому мають обмежену корисність для екстремальних явищ. У разі середнього ансамблю SREF, використовуваного в Сполучених Штатах, це зменшення корисності починається зі значень за все 0,50 дюйма (13 мм).

Прогнози кількості опадів можна перевірити кількома способами. Вимірювання дощоміру можна прив'язати до сітки середніх площадних значень, які потім порівнюються із сітками для моделей прогнозу. Оцінки метеорологічного радара можна використовувати безпосередньо або скоригувати для спостережень із дощоміром.

Деякі статистичні оцінки можуть бути засновані на полях, що спостерігаються і прогнозуються. Один, відомий як зміщення , порівнює розмір поля прогнозу з спостережуваним полем з метою отримання 1. Оцінка загрози включає перетин прогнозу і набори, що спостерігаються, з максимально можливою оцінкою верифікації 1. Імовірність виявлення , або POD, знаходиться шляхом поділу перекриття між прогнозованим і спостережуваними полями на розмір спостережуваного поля: мета тут - оцінка 1. Індекс критичного успіху, або CSI, ділить перекриття між полями прогнозу і полів, що спостерігаються на об'єднаний розмір полів прогнозу і спостережуваних: мета тут - 1 бал. частота помилкових тривог, або FAR, ділить область прогнозу, яка не перекриває поле, що

спостерігається, на розмір прогнозованої області. Цільове значення цього показника дорівнює нулю.

При тропічних циклонах, які впливають на Сполучені Штати, глобальна модель прогнозу GFS показала найкращі результати щодо прогнозів опадів протягом останніх кількох років перевершує моделі прогнозів NAM і ECMWF.

1.2. Аналіз існуючих рішень

1.2.1 EarthData сервіс

Система даних та інформації системи спостереження Землі (EOSDIS) ключова основна можливість програми НАСА по системам даних з наук про Землю. Це комплексна система даних і інформації, призначена для виконання широкого кола функцій на підтримку різнорідного національного і міжнародного співтовариства користувачів. EOSDIS надає спектр послуг; деякі сервіси призначені для різноманітної групи випадкових користувачів, в той час як інші призначені тільки для обраної групи вчених-дослідників, обраних на рецензованих конкурсах НАСА, і багато хто з них потрапляють десь посередині. Основними послугами, що надаються EOSDIS, є підтримка користувачів, архівування даних, управління та поширення, управління інформацією та створення продуктів, всі з яких знаходяться під управлінням проекту системи даних і інформації про Землю (ESDIS).

EOSDIS. обробляє, архівує та розподіляє дані з великої кількості супутників для спостереження за Землею і надає наскрізні можливості для управління даними НАСА наук про Землю з різних джерел - супутників, літаків, польові виміри і різні інші програми. Для супутникових місій системи спостереження за Землею (EOS) EOSDIS надає можливості управління і контролю, планування, збору даних і початкової (рівень 0) обробки. Ці можливості, складові операції місії EOSDIS,

управляються проектом Earth Science Mission Operations (ESMO) Project. Мережеві можливості НАСА передають дані в наукові центри. EOSDIS складається з набору засобів обробки і Центрів розподіленого активного архіву, розподілених по США. Ці кошти обробки і DAAC обслуговують сотні тисяч користувачів по всьому світу, надаючи щорічно сотні мільйонів файлів даних, що охоплюють багато дисципліни наук про Землю. Станом на вересень 2012 проект EOSDIS повідомив, що в його базі даних міститься близько 10 ПБ даних з щоденним споживанням приблизно 8,5 ТБ.

Інші можливості EOSDIS становлять наукові операції EOSDIS, які управляються проектом Системи даних та інформації про Землю (ESDIS). Ці можливості включають: створення продуктів наукових даних більш високого рівня (рівні 1-4) для місій EOS; архівування та поширення продуктів даних з EOS і інших супутникових місій, а також з літаків і польових вимірювань. Наукові операції EOSDIS виконуються в розподіленої системі з безлічі взаємозалежних вузлів (Системи обробки даних під керівництвом дослідника і розподілені, для конкретних дисциплін, Науки про Землю Розподілені центри активного архіву) з конкретними обов'язками з виробництва, архівування та поширенню продуктів даних з наук про Землю. Розподілені центри активного архівування обслуговують велику і різноманітне співтовариство користувачів (на що вказують показники продуктивності EOSDIS), надаючи можливості для пошуку і доступу до продуктів даних і спеціалізованим сервісів.

З початку 1980 по 1986 рік НАСА підтримувало пілотний проект дослідження систем даних для оцінки здійсненності і розвитку загальнодоступних електронних систем даних. Частина схвалення Конгресом місії EOS в 1990 році включала NASA Earth Science Enterprise, яка підтримала розробку системи довгострокових даних і інформації (EOSDIS). Ця система буде доступна як для науково-дослідницького співтовариства, так і для широкого загалу, побудована на розподіленої відкритій архітектурі. З урахуванням цих функціональних вимог до управління космічними

операціями і генерації продуктів для EOS, EOSDIS також відповідатиме за архівування даних, управління та поширення всіх даних приладів НАСА, що займаються дослідженням Землі, протягом всієї місії.

1.2.2 WorldWeatherOnline сервіс

У якості постачальника прогнозованих даних опадів головним чином використовуються сервіс WorldWeatherOnline з відкритим API. Local Weather API забезпечує прогноз погоди в реальному часі та на майбутнє для міст та населених пунктів по всьому світу. Він дозволяє компаніям, розробникам та програмістам вбудовувати дані про погоду у свої програми та веб-сайти.

Послуги надаються на комерційній основі, однак є можливість отримати 60-дений пробний період.

1.3. Постановка задачі

Метою розробки є дослідження оцінки точності даних прогнозованих опадів онлайн сервісами, що надають дану інформацію користувачам через зовнішнє API.

Серверний додаток використовує мову програмування та технології Java , та Spring фреймворк.

При розробці системи необхідно здійснити:

– побудову серверного додатку з можливістю отримувати прогнозовані дані шляхом відправки запитів відповідним онлайн сервісам та отримання даних від EarthData;

– побудову застосунку який надає можливість обробки даних HDF5 файлів, та надає інформацію щодо фактичної кількості опадів на відповідних встановлених координатах та часу.

РОЗДІЛ 2

МОДЕЛІ ТА МЕТОДИ РОЗВ ’ЯЗАННЯ ЗАДАЧІ

2.1. Огляд GPM сервісу

GPM – це міжнародна супутникова місія, запущена NASA та JAXA 27 лютого 2014 року, яка забезпечує спостереження нового покоління за дощем і снігом у всьому світі кожні три години. GPM (IMERG) — це уніфікований алгоритм, який надає оцінку кількості опадів, поєднуючи дані з усіх пасивних мікрохвильових приладів. Цей алгоритм призначений для взаємного калібрування, об’єднання та інтерполяції всіх супутникових оцінок (Рис. 2.1) мікрохвильових опадів разом з інфрачервоними (ІЧ) супутниковими оцінками, каліброваними в мікрохвильовому випромінюванні, аналізом датчика опадів і потенційно іншими оцінювачами опадів у точному масштабі часу та простору. Система запускається кілька разів для кожного часу спостереження, спочатку дає швидку оцінку, а потім надає кращі оцінки, коли надходить більше даних. На останньому етапі використовуються щомісячні дані вимірювання для створення продуктів на рівні досліджень.

Рис. 2.1 Ілюстрація GPM спутників

GPM забезпечує глобальні вимірювання опадів з покращеною точністю, охопленням і динамічний діапазон для вивчення характеристик опадів. Очікується, що GPM також покращиться прогнози погоди та опадів шляхом засвоєння миттєвих опадів інформації. Розширені можливості вимірювання та вибірки GPM пропонує багато передових наукових внесків і переваг для суспільства:

1. Покращені знання про кругообіг води на Землі та його зв’язок зі зміною клімату.
2. Нове уявлення про штормові структури та великомасштабні атмосферні процеси.
3. Нове розуміння мікрофізики опадів.
4. Розширене розуміння чутливості клімату та процесів зворотного зв’язку.
5. Розширені можливості моніторингу та прогнозування ураганів та інших екстремальних ситуацій (Рис. 2.2).
6. Покращені можливості прогнозування природних небезпек, включаючи повені, посуха та зсуви.
7. Покращене прогнозування сільськогосподарських культур та моніторинг ресурсів прісної води.

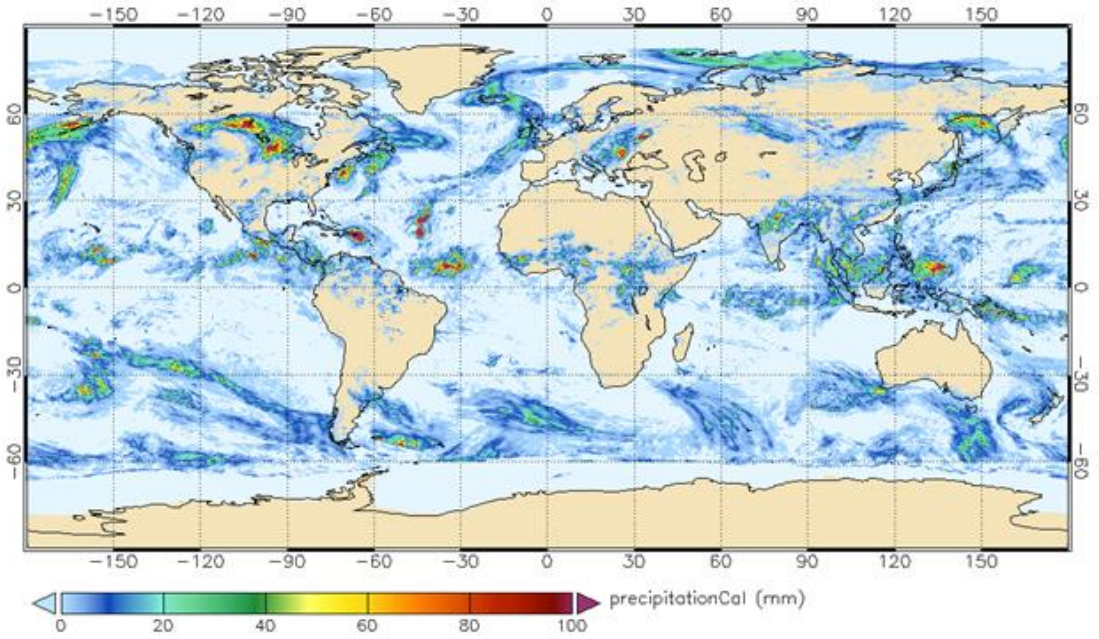


Рис. 2.2 Візуалізація опадів за GPM даними

2.2. Порівняння прогнозованих та фактичних опадів

Для порівняння точності надання прогнозу опадів Web-сервісами було обрано п’ять наступних міст з різних континентів та відповідних їм географічних координат:

1. Лондон (51.510 широта, -0.118 довгота у десяткових градусах).
2. Нью-Йорк (40.714, -74.006).
3. Сідней (-33.868, 151.207).
4. Марракеш (31.634, -7.999).
5. Токіо (35.689, 139.691).

Дані були зібрані за 2017, 2019, та 2021 роки. Увага приділялася першим двом тижням січня, квітня та вересня кожного року (табл. 2.1-2.2).

Таблиця 2.1

Приклад GPM даних опадів у міліметрах за січень 2017 року

	Місто				
Дата	Лондон	Нью-Йорк	Сідней	Марракеш	Токіо
2017.01.01	3.1	0.3	6.6	0.0	0.3
2017.01.02	0.0	6.2	2.0	0.0	0.0
2017.01.03	0.0	7.8	0.0	0.0	0.0
2017.01.04	0.0	2.1	0.2	0.0	0.0
2017.01.05	0.0	0.0	1.8	0.0	0.0
2017.01.06	5.4	0.9	1.1	0.0	0.0
2017.01.07	0.0	7.8	0.0	0.0	0.4
2017.01.08	0.0	0.0	0.1	0.1	19.7
2017.01.09	12.9	0.0	9.6	0.0	0.0
2017.01.10	0.4	0.6	0.0	0.0	0.0
2017.01.11	0.0	8.4	0.1	0.0	0.0
2017.01.12	15.8	6.3	0.0	0.0	0.0
2017.01.13	1.2	6.9	9.5	0.0	0.0
2017.01.14	0.0	1.8	0.0	0.0	0.1

Таблиця 2.2

Приклад прогнозованих WorldWeather даних опадів у міліметрах за січень 2017 року

	Місто				
Дата	Лондон	Нью-Йорк	Сідней	Марракеш	Токіо
2017.01.01	10.1	0.9	2.7	0.0	0.0
2017.01.02	2.2	2.3	0.4	0.0	0.0
2017.01.03	0.0	11.1	1.1	0.0	0.0
2017.01.04	0.0	0.0	1.2	0.0	0.0
2017.01.05	0.0	0.0	6.0	0.0	0.0
2017.01.06	2.0	4.6	6.1	0.0	0.0
2017.01.07	2.3	6.7	0.3	0.0	0.0
2017.01.08	0.0	0.3	0.0	0.0	19.2
2017.01.09	1.5	0.0	1.8	0.0	21.8
2017.01.10	0.0	0.0	1.2	0.0	0.0
2017.01.11	0.0	13.6	2.8	0.0	0.0
2017.01.12	14.8	13.4	0.4	0.0	0.0
2017.01.13	0.0	0.2	0.0	0.0	0.1
2017.01.14	0.1	2.3	1.8	0.0	0.0

2.2.1 GPM та WorldWeather API дані

У таблиці 2.3 наведено обраховані середньомісячні показники прогнозованих Web-сервісом WorldWeather та фактичними даними НАСА зібраними спутниками.

Таблиця 2.3

Середньомісячні показники опадів, мм

	Місто									
	Лондон		Нью-Йорк		Сідней		Марракеш		Токіо	
Дата	GPM	WW	GPM	WW	GPM	WW	GPM	WW	GPM	WW
2017.01	2.8	2.4	3.5	4.0	2.2	1.9	0.0	0.0	1.5	2.9
2017.04	0.3	0.5	3.1	5.4	3.6	4.5	0.0	0.1	4.0	4.3
2017.09	2.7	1.7	2.6	4.9	1.3	0.2	0.1	0.0	4.9	1.9
2019.01	1.3	0.1	2.0	3.0	3.3	2.5	0.1	0.0	0.8	0.2
2019.04	2	1.1	4.5	3.9	2.2	2.0	0.7	1.4	3.2	2.6
2019.09	1.1	0.8	4.0	3.5	0.5	0.6	3.1	4.2	10.5	11.4
2021.01	2.4	4.5	3.3	2.8	1.1	2.3	1.9	2.9	0.3	0.4
2021.04	0.4	0.5	4.2	5.0	2.4	2.5	0.1	0.2	2.8	3.7
2021.09	2.8	0.3	5.0	3.8	1.2	0.9	0.0	0.1	2.4	3.7

Враховуючи вищенаведені дані, можна стверджувати про неоднозначність точності передбачення опадів, адже кількість опадів з точністю до 20 % вдалося передбачити лише приблизно у половині випадків. Зокрема, точність передбачення зменшується зі зростанням кількості опадів, і навпаки, зі

зменшенням активності опадів середнє відхилення між даними двох джерел зменшується.

РОЗДІЛ 3

ПРОГРАМНА РЕАЛІЗАЦІЯ РОЗРОБЛЕНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1. Вибір технологій інформаційної системи

3.1.1 HDF дані

Hierarchical Data Format 5 (HDF5) - це унікальний набір технологій з відкритим вихідним кодом для керування даними колекції будь-яких розмірів і складності. HDF5 був спеціально розроблений:

- для великого обсягу та/або складних даних (але можна використовувати для малого обсягу/простих даних);
- для будь-якого розміру та типу системи (портативний) • Для гнучкого, ефективного зберігання та введення/виводу;
- дати можливість програмам розвиватися у використанні HDF5 та вміщувати нові моделі;
- використовується як набір інструментів для форматування файлів (багато форматів використовують HDF5 під капотом).

HDF5 має функції інших форматів, але він може набагато більше. У цьому HDF5 схожий на XML Файли HDF5 самоописуються і дозволяють користувачам вказувати складні зв’язки даних і залежності. На відміну від документів XML, файли HDF5 можуть містити двійкові дані (у багатьох уявлення) і надають прямий доступ до частин файлу без попереднього аналізу всього файлу зміст.

HDF5 також дозволяє ієрархічні об'єкти даних виражати природним чином (подібно до каталоги та файли), на відміну від таблиць у реляційній базі даних. Тоді як реляційні бази даних підтримують таблиці, HDF5 підтримує n-вимірні набори даних і кожен елемент у сам набір даних може бути складним об'єктом.

Реляційні бази даних пропонують чудову підтримку запитів засновані на збігу полів, але не дуже підходять для послідовної обробки всіх записів у бази даних або для вибору підмножини даних на основі пошуку в стилі координат.

3.1.2 WorldWeatherOnline дані

API локальної історичної погоди або минулої погоди (також відомий як API історичної погоди міста та міста) надає вам доступ до погодних умов з 1 липня 2008 року до теперішнього часу. API повертає такі елементи погоди, як температура, опади (опади), опис погоди, значок погоди та швидкість вітру.

Використання Web-сервісу здійснюється шляхом відправки HTTP GET запитів, де у адресі необхідно вказати наступні параметри:

- API ключ;
- географічні координати;
- необхідну дату;
- формат представлення (json).

Сервіс повертає у відповіді декілька десятків метрик (рис. 3.1), тому була необхідність розробити додатковий додаток задля поліпшення умов використання сервісу.

Додатковою є можливість отримати щомісячні середні кліматичні дані для певного місця, усереднені за останні 12 років. Доступні погодні елементи включають середні мінімальні та максимальні температури, абсолютні мінімальні та максимальні температури, середню кількість опадів та середню кількість сухих, снігових та туманних днів.


```
{
    "data": {
        "request": [
            {
                "type": "LatLon",
                "query": "Lat 51.51 and Lon -0.12"
            }
        ],
        "weather": [
            {
                "date": "2017-01-01",
                "astronomy": [
                    {
                        "sunrise": "08:06 AM",
                        "sunset": "04:02 PM",
                        "moonrise": "09:47 AM",
                        "moonset": "07:43 PM",
                        "moon_phase": "New Moon",
                        "moon_illumination": "22"
                    }
                ],
                "maxtempC": "10",
                "maxtempF": "49",
                "mintempC": "5",
                "mintempF": "40",
                "avgtempC": "7",
                "avgtempF": "45",
                "totalSnow_cm": "0.0",
                "sunHour": "3.0",
                "uvIndex": "2",
```

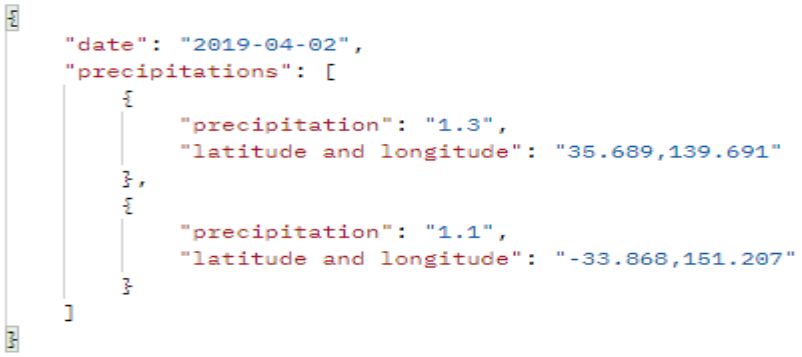

Рис. 3.1 Приклад відповіді WWO API

3.2. Створення інформаційної технології

У якості мови програмування була обрана Java. Мова програмування Java зарекомендувала себе як надійна та платформонезалежна мова, що робить її однією з кращих мов програмування для розробки enterprise-додатків - додатків масштабу підприємства. Крім того, Java – одна з основних мов для мобільної розробки, а також широко використовується для створення різноманітних Веб і десктоп додатків. Мова програмування Java зарекомендувала себе як надійна та платформонезалежна мова, що робить її однією з кращих мов програмування для розробки enterprise-додатків - додатків масштабу підприємства. Крім того, Java – одна з основних мов для мобільної розробки, а також широко використовується для створення різноманітних Веб і десктоп додатків.

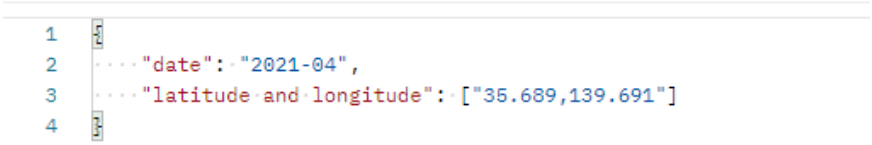
Інформаційна система включає у себе два окремих модуля:

1. для обробки HDF5 GPM даних;
2. для обробки та групування даних наданих WWO (рис.3.2 – 3.4).



```
{
    "date": "2019-04-02",
    "precipitations": [
        {
            "precipitation": "1.3",
            "latitude and longitude": "35.689,139.691"
        },
        {
            "precipitation": "1.1",
            "latitude and longitude": "-33.868,151.207"
        }
    ]
}
```


Рис.3.2 Приклад відповіді розробленого сервісу



```
{
    "date": "2021-04",
    "latitude and longitude": ["35.689,139.691"]
}
```


Рис. 3.3 Приклад запиту

```
{
    "date": "2021-04",
    "precipitations": [
        {
            "precipitation": "3.742857142857143",
            "latitude and longitude": "35.689,139.691"
        }
    ]
}
```


Рис. 3.4 Приклад відповіді для обрахування середньомісячних показників опадів

ВИСНОВКИ

Дана робота посвячена розробці інформаційної системи, що представляє собою сервіс для оцінювання точності сервісів прогнозування опадів. Акцент робиться на порівнянні фактичних даних що надає науковий проект НАСА EarthData, та прогнозованих даних наданих третьосторонніми сервісами. Проект має браузерний графічний інтерфейс користувача.

Під час виконання роботи були виконані наступні задачі:

1. Ознайомлення з актуальними методами прогнозування опадів.
2. Дослідження існуючих сервісів що надають прогнозовані дані.
3. Дослідження можливостей опрацювання даних GPM сервісу.
4. Розробка принципів оцінювання точності прогнозів.

Враховуючи опрацьовані дані, можна стверджувати про неоднозначність точності передбачення опадів, адже кількість опадів з точністю до 20 % вдалося передбачити лише приблизно у половині випадків. Зокрема, точність передбачення зменшується зі зростанням кількості опадів, і навпаки, зі зменшенням активності опадів середнє відхилення між даними двох джерел зменшується.

Одже можна стверджувати, що на сьогоднішній день середньостатистичне предствлення веб-сервісів передбачення опадів не може повною мірою бути використане у промисловості через доволі низку точність передбачення кількості опадів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Метод розрахунку деяких біокліматичних показників. Метеорологія та гідрологія / Айзенштадт Б.А. – 1964. – № 12. – С. 9–16.
2. High Level Introduction to HDF5. URL: https://support.hdfgroup.org/HDF5/Tutor/HDF5Intro.pdf (дата звернення 10.11.21)
3. Document for the GPM Data. URL: https://support.hdfgroup.org/HDF5/Tutor/HDF5Intro.pdf (дата звернення 10.11.21)
4. WorldWeatherOnline API documentation. URL: https://www.worldweatheronline.com/developer/api/docs/historical-weather-api.aspx (дата звернення 10.11.21)
5. GPM: Global Precipitation Measurement. URL: https://developers.google.com/earth-engine/datasets/catalog/NASA_GPM_L3_IMERG_V06 (дата звернення 10.11.21)
6. NASA Global Precipitation Measurement (GPM) Integrated Multi-satellitE Retrievals for GPM (IMERG). URL: https://docserver.gesdisc.eosdis.nasa.gov/public/project/GPM/IMERG_ATBD_V06.pdf (дата звернення 10.11.21)
7. The Java Tutorials. URl: https://docs.oracle.com/javase/tutorial/ (дата звернення: 10.11.2021).
8. Alex Rodriguez. RESTful Web services: The basics. URL: https://developer.ibm.com/articles/ws-restful/ (дата звернення: 22.10.2021).

ДОДАТОК А

Відомість матеріалів кваліфікаційної роботи

		Позначення	Найменування	Кільк. аркушів	Примітка
1					
2			Документація		
3					
4		**ІТКІ.КР 21.03.ДА.ПЗ**	Пояснювальна записка	52	
5					
6			Презентація	7	
7					
8			Диск CD з презентацією	1	

Зм.	Ар-куш	№ докум	Підпис	Дата	**ІТКІ.КР 21.03.ДА.ПЗ**			
Розроб.		І.А. Бородін			**Матеріали кваліфікаційної роботи**	Літ.	Аркуш	Арку-шів
						Н	1	1
Керівник		І.М. Гаркуша				НТУ «ДП», 126м-20-1		
Рецензент								
Н.контр.		Г.М. Коротенко						
Зав.каф.		В.В.Гнатушенко						

ДОДАТОК Б

Програмний код застосунку обробки HDF5

Main.java

```
package org.mygpmlib;

public class Main {
   public static void main(String[] args) {
      GPM_3IMERGDE gpm = new GPM_3IMERGDE();

      if (gpm.openGpm3ImergdeV6(${inputFilePath})) {
         if (!gpm.convertToCSV(${outputFileName}) {
            System.err.println("ERROR of convert to CSV-format!");
         }

         float p = gpm.getPrecipitationMM(longitude, latitude);
         System.out.println("Precipitation = " + p);

         gpm.closeGpm();
      }
   }
}
```

GPM_3IMERGDE.java

```
package org.mygpmlib;

import java.io.PrintWriter;
import java.io.FileNotFoundException;

import hdf.hdf5lib.H5;
import hdf.hdf5lib.HDF5Constants;
import hdf.hdf5lib.exceptions.HDF5Exception;

public class GPM_3IMERGDE {

   private static final int LONGITUDE_COUNT = 3600;
   private static final int LATITUDE_COUNT = 1800;
   private static final int NODATA = -9999;

   private long fileId = -1;
   private long datasetPrecipitationCaliD = -1;
   private long datasetLongitude = -1;
   private long datasetLatitude = -1;
   private boolean flagDataLoad = false;

   private final float[][] precipitationCal = new float[LONGITUDE_COUNT][LATITUDE_COUNT];
   private final float[] longitude = new float[LONGITUDE_COUNT];
```

```
    private final float[] latitude = new float[LATITUDE_COUNT];

    public GPM_3IMERGDE() {
    }

    public boolean openGpm3ImergdeV6(final String gpmFileName) {
        final String dsnamePCAL = "precipitationCal";
        final String dsnameLON = "lon";
        final String dsnameLAT = "lat";

        if (isOpen()) {
            closeGpm();
        }

        try {
            fileId = H5.H5Fopen(gpmFileName, HDF5Constants.H5F_ACC_RDONLY,
HDF5Constants.H5P_DEFAULT);
            if (fileId >= 0) {
                datasetPrecipitationCaliD = H5.H5Dopen(fileId, dsnamePCAL, HDF5Constants.H5P_DEFAULT);
                datasetLongitude = H5.H5Dopen(fileId, dsnameLON, HDF5Constants.H5P_DEFAULT);
                datasetLatitude = H5.H5Dopen(fileId, dsnameLAT, HDF5Constants.H5P_DEFAULT);

                if ((datasetPrecipitationCaliD >= 0) && (datasetLongitude >= 0) && (datasetLatitude >= 0)) {
                    System.out.println("Start Read Data from GPM Product...");
                    H5.H5Dread(datasetPrecipitationCaliD, HDF5Constants.H5T_NATIVE_FLOAT,
                            HDF5Constants.H5S_ALL, HDF5Constants.H5S_ALL,
                            HDF5Constants.H5P_DEFAULT, precipitationCal);

                    H5.H5Dread(datasetLongitude, HDF5Constants.H5T_NATIVE_FLOAT,
                            HDF5Constants.H5S_ALL, HDF5Constants.H5S_ALL,
                            HDF5Constants.H5P_DEFAULT, longitude);

                    H5.H5Dread(datasetLatitude, HDF5Constants.H5T_NATIVE_FLOAT,
                            HDF5Constants.H5S_ALL, HDF5Constants.H5S_ALL,
                            HDF5Constants.H5P_DEFAULT, latitude);

                    flagDataLoad = true;
                }
            }

        } catch (HDF5Exception e) {
            closeGpm();
            e.printStackTrace();
        }

        return flagDataLoad;
    }

    public boolean convertToCSV(final String csvFileName) {
        if (!isOpen()) {
            return false;
        }
```

```
        boolean res = false;

        PrintWriter fout = null;
        try {
            fout = new PrintWriter(csvFileName);
            System.out.println("Start Convert Data to ASCII file " + csvFileName + " ...");
            int oldPercent = 0;
            int percent = 0;
            fout.printf("LON;LAT;PRECIPITATION_MM\n");
            for (int y = 0; y < LATITUDE_COUNT; y++) {
                for (int x = 0; x < LONGITUDE_COUNT; x++) {
                    if (((int) (precipitationCal[x][y]) != NODATA) && (precipitationCal[x][y] != 0)) {
                        fout.printf("%f;%f;%f\n", longitude[x], latitude[y], precipitationCal[x][y]);
                    }
                }

                percent = y * 100 / LATITUDE_COUNT;
                if ((percent > oldPercent) && ((percent % 10) == 0)) {
                    System.out.println(percent + "%");
                    oldPercent = percent;
                }
            }
            System.out.println("100%\nConvert Complete!");
            fout.close();
            res = true;
        } catch (FileNotFoundException e) {
            if (fout != null) fout.close();
            e.printStackTrace();
        }
        return res;
    }

    public float getPrecipitationMM(final float lat, final float lon) {
        if (!isOpen()) {
            return NODATA;
        }

        for (int y = 0; y < LATITUDE_COUNT - 1; y++) {
            for (int x = 0; x < LONGITUDE_COUNT - 1; x++) {
                if ((longitude[x] <= lon) && (longitude[x + 1] >= lon) && (latitude[y] <= lat) && (latitude[y + 1]
>= lat)) {
                    float d1 = getDistance(longitude[x], latitude[y], lon, lat);
                    float d2 = getDistance(longitude[x + 1], latitude[y], lon, lat);
                    float d3 = getDistance(longitude[x + 1], latitude[y + 1], lon, lat);
                    float d4 = getDistance(longitude[x], latitude[y + 1], lon, lat);
                    int iDist1 = -1;
                    int iDist2 = -1;
                    int iDist = -1;
                    iDist1 = (d1 < d2) ? 0 : 1;
                    float d5 = java.lang.Math.min(d1, d2);
                    iDist2 = (d3 < d4) ? 2 : 3;
```

```
                float d6 = java.lang.Math.min(d3, d4);
                iDist = (d5 < d6) ? iDist1 : iDist2;

                switch (iDist) {
                    case 0:
                        return getPrecipitation(x, y);
                    case 1:
                        return getPrecipitation(x + 1, y);
                    case 2:
                        return getPrecipitation(x + 1, y + 1);
                    case 3:
                        return getPrecipitation(x, y + 1);
                }
            }
        }
    }

    return NODATA;
}

public void closeGpm() {
    try {
        if (datasetPrecipitationCaliD >= 0) {
            H5.H5Dclose(datasetPrecipitationCaliD);
        }
        if (datasetLongitude >= 0) {
            H5.H5Dclose(datasetLongitude);
        }
        if (datasetLatitude >= 0) {
            H5.H5Dclose(datasetLatitude);
        }
        if (fileId >= 0) {
            H5.H5Fclose(fileId);
        }
        flagDataLoad = false;
    } catch (HDF5Exception e) {
        flagDataLoad = false;
        e.printStackTrace();
    }
}

public boolean isOpen() {
    return flagDataLoad;
}

private float getDistance(final float lon1, final float lat1, final float lon2, final float lat2) {
    return (float) java.lang.Math.sqrt((lon2 - lon1) * (lon2 - lon1) + (lat2 - lat1) * (lat2 - lat1));
}

private float getPrecipitation(int x, int y) {
    if ((int) (precipitationCal[x][y]) != NODATA) {
        return precipitationCal[x][y];
```

```
        } else {
            return NODATA;
        }
    }
}
```

ДОДАТОК В

Програмний код застосунку обробки даних сервісів передбачення опадів

Application.java

```
package com.nmu.diploma;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class Application {

    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }

}
```

PropertiesProvider.java

```
package com.nmu.diploma.configuration;

import lombok.Getter;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.context.annotation.Configuration;

@Configuration
@Getter
public class PropertiesProvider {

    @Value("${worldweatheronline.api.url}")
    private String worldWeatherOnlineApiUrl;

    @Value("${worldweatheronline.api.key}")
    private String worldWeatherOnlineApiKey;

    @Value("${worldweatheronline.api.format}")
    private String worldWeatherOnlineApiFormat;

    @Value("${worldweatheronline.api.tp}")
    private String worldWeatherOnlineApiTp;
}
```

PrecipitationController.java

```
package com.nmu.diploma.controller;

import com.nmu.diploma.dto.PrecipitationRequestDTO;
import com.nmu.diploma.dto.PrecipitationResponseDTO;
import com.nmu.diploma.service.PrecipitationService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
@RequestMapping("/api/v1/precipitation")
public class PrecipitationController {

    @Autowired
    private PrecipitationService precipitationService;

    @GetMapping
    public PrecipitationResponseDTO getPrecipitationData(@RequestBody PrecipitationRequestDTO
requestDTO) {
        return precipitationService.getPrecipitationData(requestDTO);
    }

    @GetMapping("/month")
    public PrecipitationResponseDTO getPrecipitationDataAvgMonth(@RequestBody
PrecipitationRequestDTO requestDTO) {
        return precipitationService.getPrecipitationDataAvgHalfMonth(requestDTO);
    }
}
```

Precipitation.java

```
package com.nmu.diploma.domain;

import com.fasterxml.jackson.annotation.JsonIgnore;
import com.fasterxml.jackson.annotation.JsonProperty;
import lombok.Builder;
import lombok.Data;

@Builder
@Data
public class Precipitation {

    @JsonIgnore
    private String date;

    @JsonProperty("latitude and longitude")
    private String latitudeAndLongitude;
```

```
    private String precipitation;
}
```

PrecipitationRequestDTO.java

```
package com.nmu.diploma.dto;

import com.fasterxml.jackson.annotation.JsonProperty;
import lombok.Data;

import java.util.List;

@Data
public class PrecipitationRequestDTO {
    private String date;

    @JsonProperty("latitude and longitude")
    private List<String> latitudeAndLongitude;
}
```

PrecipitationResponseDTO.java

```
package com.nmu.diploma.dto;

import com.nmu.diploma.domain.Precipitation;
import lombok.Builder;
import lombok.Data;

import java.util.List;

@Builder
@Data
public class PrecipitationResponseDTO {

    private String date;
    private List<Precipitation> precipitations;
}
```

WorldWeatherOnlineResponseDTO.java

```
package com.nmu.diploma.dto;

import com.fasterxml.jackson.annotation.JsonIgnoreProperties;
import com.fasterxml.jackson.annotation.JsonProperty;
import lombok.Data;

import java.util.List;

@Data
```

```
public class WorldWeatherOnlineResponseDTO {

   @JsonProperty("data")
   private Data data;

   @lombok.Data
   @JsonIgnoreProperties(ignoreUnknown = true)
   public static class Data {
      @JsonProperty("weather")
      private List<Weather> weather;
   }

   @lombok.Data
   @JsonIgnoreProperties(ignoreUnknown = true)
   public static class Weather {
      @JsonProperty("hourly")
      private List<Hourly> hourly;
   }

   @lombok.Data
   @JsonIgnoreProperties(ignoreUnknown = true)
   public static class Hourly {
      private String precipMM;
   }

}
```

PrecipitationService.java

```
package com.nmu.diploma.service;

import com.nmu.diploma.dto.PrecipitationRequestDTO;
import com.nmu.diploma.dto.PrecipitationResponseDTO;

public interface PrecipitationService {

   /**
    *
    * @param requestDTO request body, which contains date (2009-02-21 format) and list of coordinates
(latitude and longitude)
    * @return structure with defined precipitation for corresponding coordinates
    */
   PrecipitationResponseDTO getPrecipitationData(PrecipitationRequestDTO requestDTO);

   /**
    *
    * @param requestDTO request body, which contains date (2009-02 format) and coordinates (latitude and
longitude)
    * @return structure with defined precipitation for corresponding coordinates
    */
```

```
    PrecipitationResponseDTO getPrecipitationDataAvgHalfMonth(PrecipitationRequestDTO requestDTO);
}
```

PrecipitationServiceImpl.java

```
package com.nmu.diploma.service;

import com.fasterxml.jackson.databind.ObjectMapper;
import com.nmu.diploma.configuration.PropertiesProvider;
import com.nmu.diploma.domain.Precipitation;
import com.nmu.diploma.dto.PrecipitationRequestDTO;
import com.nmu.diploma.dto.PrecipitationResponseDTO;
import com.nmu.diploma.dto.WorldWeatherOnlineResponseDTO;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.io.IOException;

import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.URL;

import java.util.*;

@Service
public class PrecipitationServiceImpl implements PrecipitationService {

    private static final String REQUEST_PARAMETER_KEY = "key";
    private static final String REQUEST_PARAMETER_DATE = "date";
    private static final String REQUEST_PARAMETER_FORMAT = "format";
    private static final String REQUEST_PARAMETER_TP = "tp";
    private static final String REQUEST_PARAMETER_COORDINATES = "q";

    @Autowired
    private PropertiesProvider properties;

    @Override
    public PrecipitationResponseDTO getPrecipitationData(PrecipitationRequestDTO requestDTO) {
        return PrecipitationResponseDTO.builder()
                .date(requestDTO.getDate())
                .precipitations(generatePrecipitations(requestDTO))
                .build();
    }

    @Override
    public PrecipitationResponseDTO getPrecipitationDataAvgHalfMonth(PrecipitationRequestDTO
requestDTO) {
        return PrecipitationResponseDTO.builder()
                .date(requestDTO.getDate())
                .precipitations(generatePrecipitationsAvgMonth(requestDTO))
                .build();
```

```
    }

    private List<Precipitation> generatePrecipitations(PrecipitationRequestDTO requestDTO) {
        List<Precipitation> precipitations = new ArrayList<>();
        for (String latitudeAndLongitude : requestDTO.getLatitudeAndLongitude()) {
            String precipitationFromResponse =
getPrecipitationFromWorldWeatherResponse(latitudeAndLongitude,
                    requestDTO.getDate());
            Precipitation precipitation = Precipitation.builder()
                    .date(requestDTO.getDate())
                    .latitudeAndLongitude(latitudeAndLongitude)
                    .precipitation(precipitationFromResponse)
                    .build();
            precipitations.add(precipitation);
        }
        return precipitations;
    }

    private List<Precipitation> generatePrecipitationsAvgHalfMonth(PrecipitationRequestDTO requestDTO)
{
        String latitudeAndLongitude = requestDTO.getLatitudeAndLongitude().get(0);
        List<Precipitation> precipitations = new ArrayList<>();
        List<String> precipitationsPerDayOfHalfMonth = new ArrayList<>();

        for (int x = 1; x <= 14; x++) {
            String date = requestDTO.getDate() + "-" + x;
            String precipitationFromResponse =
getPrecipitationFromWorldWeatherResponse(latitudeAndLongitude, date);
            precipitationsPerDayOfHalfMonth.add(precipitationFromResponse);
        }

        OptionalDouble average = precipitationsPerDayOfHalfMonth
                .stream()
                .mapToDouble(Double::parseDouble)
                .average();
        String precipitationAvgMonth = String.valueOf(average.getAsDouble());

        Precipitation precipitation = Precipitation.builder()
                .date(requestDTO.getDate())
                .latitudeAndLongitude(latitudeAndLongitude)
                .precipitation(precipitationAvgMonth)
                .build();
        precipitations.add(precipitation);
        return precipitations;
    }

    private String getPrecipitationFromWorldWeatherResponse(String latitudeAndLongitude, String date) {
        String worldWeatherApiRequestUrl = properties.getWorldWeatherOnlineApiUrl() + "?"
                + REQUEST_PARAMETER_KEY + "=" + properties.getWorldWeatherOnlineApiKey()
                + "&" + REQUEST_PARAMETER_COORDINATES + "=" + latitudeAndLongitude
                + "&" + REQUEST_PARAMETER_DATE + "=" + date
                + "&" + REQUEST_PARAMETER_FORMAT + "=" +
```

```
properties.getWorldWeatherOnlineApiFormat()
            + "&" + REQUEST_PARAMETER_TP + "=" + properties.getWorldWeatherOnlineApiTp();

        URL url;
        HttpURLConnection connection;
        InputStream responseStream = null;
        WorldWeatherOnlineResponseDTO response = null;
        ObjectMapper mapper = new ObjectMapper();
        try {
            url = new URL(worldWeatherApiRequestUrl);
            connection = (HttpURLConnection) url.openConnection();
            connection.setRequestProperty("accept", "application/json");
            responseStream = connection.getInputStream();
            response = mapper.readValue(responseStream, WorldWeatherOnlineResponseDTO.class);
        } catch (IOException ex) {
            System.err.println(ex);
        }

        if (Objects.nonNull(response)) {
            return response.getData().getWeather().get(0).getHourly().get(0).getPrecipMM();
        }
        return "";
    }
}
```

application.properties

```
worldweatheronline.api.url=http://api.worldweatheronline.com/premium/v1/past-weather.ashx
worldweatheronline.api.key=cd64abe8099144b7844201452221801
worldweatheronline.api.format=json
worldweatheronline.api.tp=24
```

ДОДАТОК Г
ВІДГУК

ДОДАТОК Д
РЕЦЕНЗІЯ